

AD-A181 531

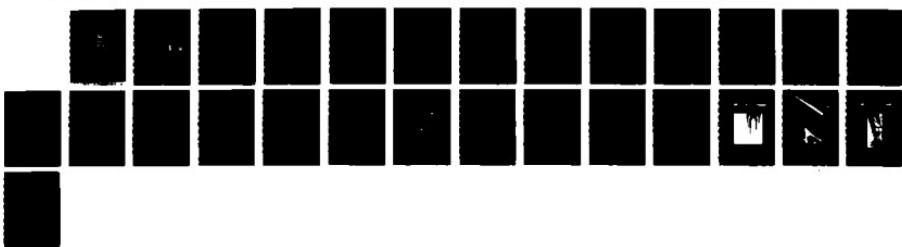
DOCUMENTATION FOR THE SHADO PARTICLE WAKE ROUTINE(U)
S-CUBED LA JOLLA CA D L PETERKA ET AL JAN 87
SSS-R-87-8495 AFGL-TR-87-0042 F19668-86-C-0056

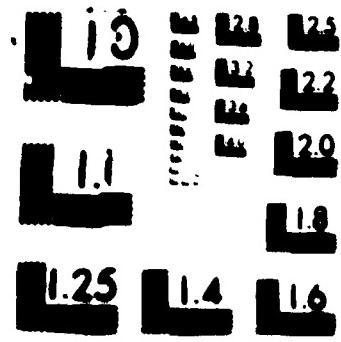
1/1

UNCLASSIFIED

F/G 22/2

NL





AD-A181 531

DTIC FILE COPY

(2)

AFGL-TR-87-0042

**Documentation for the Shado
Particle Wake Routine**

**D. L. Peterka
I. Katz**

**S-CUBED
A Division of Maxwell Laboratories
P.O. Box 1620
La Jolla, CA 92038**

January 1987

Scientific Report No. 5

Approved for Public Release; Distribution Unlimited

**AIR FORCE GEOPHYSICS LABORATORY
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
HANSOM AIR FORCE BASE
MASSACHUSETTS 01731**

DTIC
ELECTED
JUN 19 1987
S D
L D

"This technical report has been reviewed and is approved for publication"



DAVID COOKE
Contract Manager
Spacecraft Interactions Branch
Space Physics Division



CHARLES P. PIKE, Chief
Spacecraft Interactions Branch
Space Physics Division

FOR THE COMMANDER



RITA C. SAGALYN, Director
Space Physics Division

This report has been reviewed by the ESD Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS).

Qualified requestors may obtain additional copies from the Defense Technical Information Center. All others should apply to the National Technical Information Service.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify AFGL/DAA, Hanscom AFB, MA 01731. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

A181531

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b RESTRICTIVE MARKINGS										
2a SECURITY CLASSIFICATION AUTHORITY N/A		3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.										
2b DECLASSIFICATION/DOWNGRADING SCHEDULE N/A												
4 PERFORMING ORGANIZATION REPORT NUMBER(S) SSS-R-87-8495		5 MONITORING ORGANIZATION REPORT NUMBER(S) AFGL-TR-87-0042										
6a NAME OF PERFORMING ORGANIZATION S-CUBED, A Division of Maxwell Labsortories, Inc.	6b OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION Air Force Geophysics Laboratory										
6c ADDRESS (City, State, and ZIP Code) P.O. Box 1620 La Jolla, CA 92038		7b. ADDRESS (City, State, and ZIP Code) Hanscom Air Force Base Massachusetts 01731										
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Geophysics Laboratory	8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F19268-86-C-0056										
8c. ADDRESS (City, State, and ZIP Code) Hanscom Air Force Base, MA 01731		10 SOURCE OF FUNDING NUMBERS <table border="1"><tr><td>PROGRAM ELEMENT NO 62101F</td><td>PROJECT NO 7601</td><td>TASK NO 30</td><td>WORK U.N.T. ACCESSION NO AA</td></tr></table>		PROGRAM ELEMENT NO 62101F	PROJECT NO 7601	TASK NO 30	WORK U.N.T. ACCESSION NO AA					
PROGRAM ELEMENT NO 62101F	PROJECT NO 7601	TASK NO 30	WORK U.N.T. ACCESSION NO AA									
11 TITLE (Include Security Classification) Documentation for the SHADO Particle Wake Routine												
12 PERSONAL AUTHOR(S) Peterka, D. and I. Katz												
13a TYPE OF REPORT Technical Report No. 5	13b TIME COVERED FROM 10/86 TO 1/87	14 DATE OF REPORT (Year, Month, Day) January 1987	15 PAGE COUNT 24									
16 SUPPLEMENTARY NOTATION												
17 COSATI CODES <table border="1"><tr><th>FIELD</th><th>GROUP</th><th>SUB-GROUP</th></tr><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>		FIELD	GROUP	SUB-GROUP							18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Spacecraft Charging; Particle Wake; 3-B; POLAR Computer Code	
FIELD	GROUP	SUB-GROUP										
19 ABSTRACT (Continue on reverse if necessary and identify by block number) This report documents the computational algorithms of the SHADO routine for computing the particle wake behind large spacecraft in low polar orbit. SHADO will replace the existing module for computing particle densities in the POLAR code and achieves a significant improvement in computational speed.												
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT <input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED										
22a NAME OF RESPONSIBLE INDIVIDUAL David L. Cooke		22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL AFGL/PHK									

SECURITY CLASSIFICATION OF THIS PAGE

SECURITY CLASSIFICATION OF THIS PAGE

1 INTRODUCTION

SHADO is a new module for use in computing the particle wake behind an object in Low Earth Orbit (LEO) traveling at mesosonic speeds. At mesosonic velocities, a spacecraft will sweep away ambient ions since it is moving faster than the ion thermal velocity and much slower than the electron thermal velocity. This condition is encountered in LEO where the night side plasma can have a Debye length of 2 cm while the ion Debye length is roughly one fifth the spacecraft dimension. The motion of ions to fill in the electron rich wake behind the spacecraft is quickly dominated by the electric field in the wake which is obtained by solving Poisson's equation [1].

Two simplifying assumptions are made in computing the ion densities in the wake. The first assumption is the neutral ion approximation which assumes ions travel in straight lines and are stopped if they impact the spacecraft. This neglects electric field effects on ion trajectories. The second assumption is that ion filling of the wake is due to electric field acceleration, thereby neglecting ion momentum. The neutral ion approximation is expressed by the following equation:

$$f_i(\vec{x}, \vec{v}) = g(\vec{x}, \Omega) f_{io}(\vec{v}) \quad (1)$$

where $f_i(\vec{x}, \vec{v})$ is the ion distribution function at a point \vec{x} in space for a velocity \vec{v} and $f_{io}(\vec{v})$ is the unperturbed velocity distribution function for a drifting Maxwellian. The function $g(\vec{x}, \Omega)$ has a value of zero if a ray starting from \vec{x} in direction Ω would strike the spacecraft; otherwise it is one. The charge density is obtained by integration over velocities:

$$n_i(\vec{x}) = \int f_i(\vec{x}, \vec{v}) = \int g(\vec{x}, \Omega) \left[\int_0^\infty f_{io}(\nu, \Omega) \nu^2 d\nu \right] d\Omega \quad (2)$$

The object being studied is described by a collection of surface elements which are referred to as object definition surface elements. In order to determine the charge density at a point in space, it is necessary to determine the solid angle which is occupied by the object and performing the integration in equation 2. The problem is reduced to finding $g(\vec{x}, \Omega)$ given \vec{x} due to the neutral ion approximation.

The older algorithm computed the charge density for a given point in space by determining the solid angle obstructed by each surface in the object

CONFIDENTIAL

Drift	Avail / d / or Special
A-1	

definition obstructed when viewed from \vec{r} . This was accomplished by defining a spherical grid centered at \vec{r} whose $\theta = 0, \phi = 0$ ray pointed directly into the oncoming ions. The grid was subdivided into 1 degree elements in the polar angle θ , and 10 degree elements in ϕ . The greater resolution in θ was required since the function f_{io} changes rapidly as the polar angle increases. Each surface was then projected onto this spherical grid and a number of extra vertices added to account for the curvature induced by the projection of straight lines onto a curved surface. Each element in the sphere which was covered by the projected surface element was then given a value of $g_{\theta,\phi} = 0$ and integrated over θ and ϕ to obtain the charge density. [2]

While this approach was practical for simple objects, it became clear that complex objects with many surface elements would be increasingly difficult to handle. This is especially true when several million \vec{r} coordinates are required to adequately compute the particle wake behind a spacecraft. Another application for which this approach is inadequate is problem of sheath shadowing. The sheath around the moving object can be described using surface elements and treated as an object. However, very many surface elements are required and the old algorithm is too cumbersome to compute both the particle wake behind the object and the sheath shadowing.

A new approach was then developed which greatly simplifies the object definition so that the charge density may be computed much more rapidly. The new algorithm also relies heavily on the fact that the ion distribution, relative to the spacecraft, is heavily weighted towards the ram direction at mesosonic velocities. Since f_{io} is azimuthally symmetric, equation 2 can be reduced to:

$$GI(\vec{r}) = \sum_{\theta_i} f_{cum}(\theta_i) \left(\sum_{\phi_i} g(\theta_i, \phi_i) \right) \Delta\theta_i, \quad (3)$$

where the GI 's are called 'geometrical ions' obtained by dividing equation 2 by the unperturbed ion density (n_{io}). If a point \vec{r} is completely blocked by the spacecraft, GI will be zero, while GI will be unity far from the spacecraft.

The function $f_{cum}(\theta)$ is obtained by integrating $f_{io}(\theta, \phi, |\vec{v}|)$ over ϕ since it is azimuthally symmetric, and normalizing it as follows:

$$f_{cum}(\theta) = \sum_{\phi_i} f_{io}(\theta_i, \phi_i, v) \Delta\phi_i, \quad (4)$$

$$\sum_{\theta_i} f_{cum}(\theta_i) \Delta\theta_i = 1 \quad (5)$$

f_{cum} is calculated in advance, given \vec{v} , and is stored internally as an array which rapidly decreases as θ increases for mesosonic velocities. The value of θ at which f_{cum} drops to 0.01% of its value in the ram direction ($\theta = 0$) is computed and saved as $\theta_{99\%}$. As $|\vec{v}|$ increases, $\theta_{99\%}$ decreases indicating that the Maxwellian distribution of ions is shifted towards the ram direction. Surfaces lying within $0 \leq \theta \leq \theta_{99\%}$ from the perspective of \vec{x} will therefore have a significant effect on the value of $GI(\vec{x})$, while surfaces positioned beyond $\theta_{99\%}$ will have a negligible impact.

Simplification of the object definition is achieved by first projecting the object onto a surface which is normal to the ram direction and then overlaying this projection with a new two dimensional grid as illustrated in Figure 4. The nodes on this grid are arranged in an equilaterally spaced staggered array resulting in hexagonal elements. These hexagonal elements provide greater angular resolution of the object projection than an equivalent cartesian grid system. Each surface element in the object definition is then sequentially projected onto this hexagonal grid and a depth is computed for each grid node which falls inside the surface element projection. This depth represents the distance from the projection surface to the surface element along a ray parallel to the ram direction. The projection surface is placed so that the forward-most point on the object corresponds to zero depth. All of these calculations are performed only once prior to evaluation of any GI 's.

The number of nodes making up the hexagonal mesh is controlled by a parameter N_{hex} . The nodes are distributed uniformly over the rectangular area on the projection plane which just encompasses the entire object projection. As each object definition surface is processed, a table of depths for each node in the hexagonal mesh is maintained. After all the surfaces are projected, the table for each node is sorted by increasing depth. Thus, at each node on the hexagonal grid there is a table of depths representing the intersection of a ray beginning at the node and any of the object definition surfaces in the path along the anti-ram direction. Given a \vec{x} coordinate, it is then simple to compute its distance from the projection plane and its location on the hexagonal grid. Comparing the depth of \vec{x} to the table of depths of the adjacent hexagonal grid nodes it is quickly determined whether \vec{x} is in front of, behind, inside, or adjacent to the object.

If the reference point, \vec{x} , is in front of the object, then $GI(\vec{x})$ will be unity because all angles θ pointing to the object definition surfaces will be much larger than $\theta_{99\%}$. If the reference point is sufficiently far away from

the object such that the entire object lies outside of the $\theta_{99\%}$ limit, then GI is set to unity as well. If the reference point is inside the object, then GI is set to zero. Since many of the points in the object mesh fall into one of these categories, little computational effort is expended in determining the $GI(\vec{x})$'s. For the remaining points in the object mesh, it is necessary to determine where the object is relative to \vec{x} and determine how much of the view towards the \vec{x} direction is blocked.

Much of the computational speed improvement in the SHADO module is attributable to the simplification of the object representation already described. Given an \vec{x} coordinate, say directly behind the object, a scan along a number of rays controlled by a parameter N_{phi} , is initiated. Each ray begins at \vec{x} and proceeds along a fixed angle ϕ in the plane of the projection surface. At intervals just less than the hexagonal mesh element size, the surrounding hexagonal nodes are tested to determine whether they are over the object or not. If the table of depths for one or more of these nodes is empty, then this indicates that the scan has proceeded beyond the projection of the object. A quick calculation is made which approximates the distance to the object outline (accurate to half the hexagonal mesh dimension) and the depth to the object at this point is computed. This gives the angle θ from \vec{x} to the boundary of the object and the contribution to $GI(\vec{x})$ along this value of ϕ is determined. Summing for each of the N_{phi} values of ϕ gives the value the geometrical ion at \vec{x} .

A number of variations on the scheme outlined above are possible based on whether the point \vec{x} is directly behind (the table of depths on all nodes adjacent to \vec{x} are non-zero), or off to one side of the object. The same basic procedure is followed however, by scanning in rays along the projection plane, determining where the object is relative to \vec{x} on this plane, and then determining the angles, θ , locating the boundaries of the object in three dimensions relative to \vec{x} . The accuracy of the object description using this algorithm is controlled by the number of hexagonal grid nodes (N_{hex}) placed on the hexagonal grid and the number of scan angles (N_{phi}) used when locating the object relative to \vec{x} .

There are also a number of shortcuts used in the implementation of this algorithm. The major one being that the hexagonal grid is skewed such that the y axis is at a 60 degree angle with the x axis on the projection plane. This results in a cartesian spacing of the grid nodes, which when normalized to the distance between grid points, puts the skewed grid nodes

on integer coordinates. During the SHADO setup, the object definition surface elements are actually projected onto this skewed coordinate system, and all of the scans along lines of constant ϕ take place in this coordinate system as well. In the example problems which follow, the objects are shown projected on this skewed mesh with a resulting distortion of the object.

2 SAMPLE CALCULATIONS

A number of test cases have been run to determine the accuracy limits, the computational speed improvements, and to compare results against the older algorithm. In particular, comparisons were made for a 24 surface quasisphere, a 200 surface shuttle orbiter, and a 1700 surface representation of the National Aerospaceplane. The quasisphere was used to investigate accuracy limits for varying values of N_{hex} (the hexagonal grid resolution) and N_ϕ (the angular resolution in the hexagonal grid). The shuttle orbiter model was to demonstrate the speed improvements over the older algorithm, and the National Aerospaceplane model served as a demonstration of the flexibility of this new algorithm for problems of great complexity.

2.1 Quasisphere Results

The quasisphere is a simple object made up of 24 surfaces which approximate a sphere with a diameter of 12. Calculations for both the new and old algorithms consisted of calculating the density along a line 5 radii behind the sphere and perpendicular to the ram direction. The line begins roughly one diameter off the axis of symmetry and extends to the axis of symmetry. Figure 5 shows the results obtained for this problem from the old algorithm. The curve represents the drop in GI as a traverse is made from 20 units off the centerline to the centerline. The quasisphere is centered along the $x=10$ value on the figure. The Mach number used was 8.1 and showed good agreement with the experimental results of Fournier and Pigache [1].

Figure 6 shows the projection of the quasisphere on the hexagonal mesh used in the new algorithm. The grid size is a function of the N_{hex} parameter and is set to 1000 which means that the hex grid is scaled such that no more than 1000 elements are used to contain the projection of the object. Depending on the aspect ratio of the object projection, the number of elements will generally be less than N_{hex} by a small amount. The stars in figure 6

represent the vertices of a three dimensional box which just contains the object. Due to the direct alignment of the quasisphere with respect to the ram direction, four of the eight vertices of this box are stacked directly on top of the other four.

Figures 7-10, show the dependency of the solution on the angular resolution with N_ϕ set to 8, 16, 32 and 64 respectively. The effect of increasing angular resolution is a smoothing out of the density contour. The bump in figure 7 and to a lesser extent in figure 8, is caused by the interception of a single scan angle when it first finds that it is blocked by the object.

Another study of the sensitivity to the N_{hex} parameter for grid resolution showed no improvements for values higher than the value of 1000 shown in figure 6, although the density contour roughness was slightly reduced as N_{hex} was increased. A general rule of thumb would be to set N_{hex} to adequately resolve the smallest feature of interest on the object.

2.2 Shuttle Orbiter Results

In order to compare the speed of execution for both the new and the old algorithms, a 3-D calculation of the ion density around the shuttle orbiter was run. Figure 11 shows the shuttle object definition and its projection on the hexagonal mesh. In the wake calculation shown in Figure 12, the orbiter is oriented with its bay doors open and in the ram direction. The value of N_{hex} was 1000 and the value for N_ϕ was 16. Figure 8 shows the resulting particle wake through a section taken from approximately midships (the nose of the orbiter points into the page, the wings are vertical). The coarseness in the density contours is due to the low resolution used in selecting \bar{x} values. The Mach number was 8.0 for this problem and N_ϕ was set to 16.

The speed of the SHADO algorithm was clearly demonstrated in this example. The old algorithm required roughly 30 times more CPU time to calculate ion density for the roughly 100,000 points than did the SHADO algorithm. On a Ridge 32 computer, the computation time was reduced from approximately 30 hours to under one hour.

2.3 National Aerospaceplane Results

As a demonstration of the versatility of this new algorithm, a particle wake was computed for a detailed rendition of the National Aerospaceplane, described by over 1700 surface elements. The angle of attack was 20 degrees at Mach 8. Figure 13 shows the object definition used and the projection onto the hexagonal mesh. Figure 14 shows the particle wake behind the aerospaceplane taken though the midplane and includes a silhouette of the object. This calculation required N_{hex} to be set at 5000 in order to adequately capture the particle wake at the sharp nose. N_ϕ was set to 64 and a dense spatial resolution in \bar{z} was used in order to obtain smooth density contours. The calculation shown in the figure represents over 10,000 spatial coordinates and required a little over 7 minutes of CPU time on a Ridge 32 computer.

This example demonstrates the strength of this new algorithm in its ability to deal with complex objects in an efficient manner. The SHDSET setup routine first reduces the problem from a collection of many surfaces to a projected surface of order N_{hex} elements. Once this preprocessing is accomplished, the algorithm is constrained only by the mesh size as dictated by the N_{hex} parameter and the angular resolution as dictated by the N_ϕ parameter. The computational efficiency of the old algorithm at each point in space was directly linked to the number of surfaces used in the object representation which often required that complex objects be simplified by combining surface elements into larger elements.

3 SHADO STRUCTURE

The SHADO routines are organized in a highly modular fashion. The following sections detail the organization of the routines and describes their function. The overall structure diagram for SHADO is shown in Figure 1. The driver is the POLAR code. SHDSET is the routine which processes the object definition surface elements, and SHADO is the routine which computes the GI given \bar{z} .

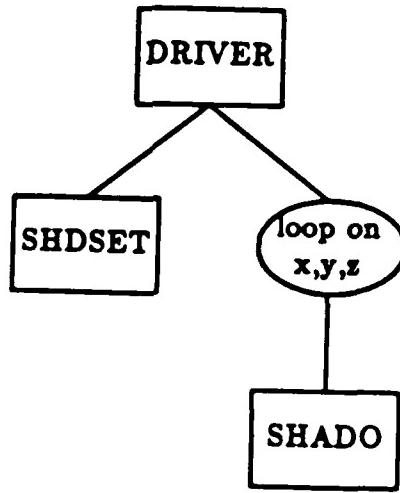


Figure 1: Shado Structure Diagram

3.1 SHDSET - Set Up Routine

Initialization for the SHADO routine takes place in SHDSET and consists of looping over each surface defining an object, projecting that surface onto the plane perpendicular to the ram direction, and determining which nodes on a hexagonal grid are shaded by each surface. For each surface, limits are determined which define a box in the hex mesh which encloses the projected surface. Only points inside this box are checked for shadowing by that surface.

At each mesh point in the hexagonal grid, a record is kept of the height of intersection from the projection surface and the number of surfaces which shade a given mesh point. An array stores the z values of each surface intersection sorted by ascending node number and then by ascending z value. Given any node, it is then simple to determine if this node is shaded in the ram direction and to find the entry and exit z values.

The structure diagram for SHDSET is shown in Figure 2.

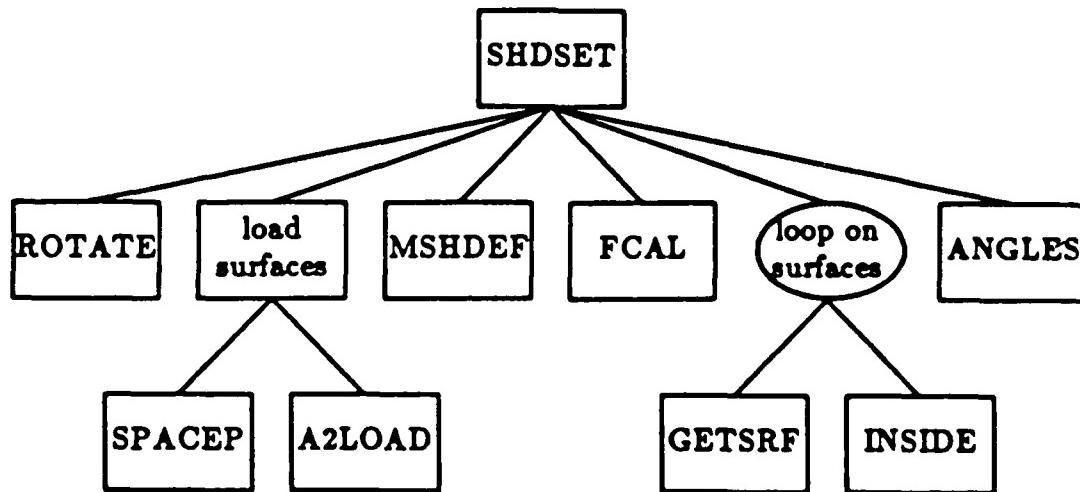


Figure 2: SHDSET structure diagram

3.1.1 ROTATE

Computes the rotation matrix required to convert any (x,y,z) to (x',y',z') in which the x'-y' plane is normal to the Mach vector. The x' axis is taken to be in the x-z plane.

3.1.2 SPACEP & A2LOAD

Reads in surface element data for the National aerospace plane using the MSIO library. Alternatively, A2 surfaces may be read using routines A2LOAD and A2PREP. These routines simply read in data from MSIO data files and load an array of vertices for each surface. These routines will also find the minimum and maximum x,y, and z values within the object to be used in defining the hexagonal mesh. Data selection is determined by the LUSURF parameter which corresponds to the logical unit number for the surface data 19 for the A2's, 9 for the spaceplane data. Surfaces are defined by coordinates for surface vertices and an outward pointing normal vector. This form is exactly how A2 surfaces are defined, for the spaceplane, however, the normal is computed using three distinct vertices to compute the normal vector.

assuming clockwise node numbering

3.1.3 MSHDEF

Using the minimum and maximum x, y, and z values, a box is defined within which the object is fully contained. The eight vertices making up this box are then projected onto the x' y' plane and the minimum and maximum x' and y' values are found. A hexagonal mesh is then established which has its origin at $(x_{\min} - \Delta, y_{\min} - \Delta)$ and extends up to $(x_{\max} + \Delta, y_{\max} + \Delta)$ where Δ is the mesh spacing. Δ is chosen using iteration such that the number of mesh points is $\leq N_{hex}$. The mesh is then skewed 60 degrees to give a square mesh and normalized by Δ such that the nodes on the mesh are integer values.

3.1.4 FCAL

Calculates the ion density factor array for a point in space which is unobstructed in all directions. The ion density factor is computed as a function of the out of plane angle θ as follows

$$f(\theta) = \frac{4\sqrt{2}e^{-y^2}}{\sqrt{2}}$$

where

$$y = V_{max} \frac{\cos \theta}{\sqrt{2}}$$

$$A = \begin{cases} \sqrt{\pi} e^{y^2} (1 - erfc(y)) & y \geq 0 \\ y^{-1} (1 - \frac{1}{2} y^2 + \frac{1}{8} y^4 - \frac{1}{128} y^6 + \dots) & y < 0 \end{cases}$$

The $f(\theta)$'s are calculated over a uniform grid with $\Delta\theta = 10^\circ$, i.e. 36 degrees in θ and then integrated over θ to give

$$f_{sum,\theta} = \int_0^{\pi} f(\theta) d\theta$$

3.1.5 GETSRF

Reads in the surfaces of the object and returns the internal limits for the hexagonal mesh points which must be checked for shadowing. Any surface which is perpendicular to the projection surface is ignored. A loop inside SHDSET reads in the number of surfaces from this file, and GETSRF gets a valid surface. The return arguments are the coordinates of up to four vertices making up a surface element which is not perpendicular to the x'y' plane.

3.1.6 INSIDE

Fills an array which retains all of the surface intersections at each hexagonal grid point which is shaded by a surface. For each mesh point which is blocked by the given surface, the height of the surface above the projection surface is computed and stored in an array of z's and the corresponding node number is stored in a companion array. After all of the surfaces have been processed, these two arrays are sorted by node in ascending order of z' to obtain the minimum z value (z_{min}) and maximum z value (z_{max}) for each of the blocked nodes.

3.1.7 ANGLES

In SHADO, the hexagonal mesh is skewed by 60 degrees to become a cartesian grid system. Scanning to define the object boundary takes place over a given number of angles in the x'y' plane, which when skewed, become lines which are no longer uniformly spaced angularly. This routine computes the increments in x' and y' to be taken along each of these lines. The increments are selected to be 99 unskewed mesh

$$\Delta x_i = 0.99 (\sqrt{3} \sin \phi_i + \cos \phi_i) \quad (10)$$

$$\Delta y_i = 0.99 \cos \phi_i \quad (11)$$

Where ϕ is the angle in the x'y' plane measured from the x' axis and incremented by $\phi = i \cdot \frac{\pi}{N_s}$. N_s is the number of scan angles to be used in the x'y' plane.

3.2 SHADO - Calculate Geometrical Ion Density

SHADO calculates the presheath ion density using the neutral ion approximation. This approximation assumes that ion motion is perturbed only by collisions with an absorbing object. The ion density accounts fully for ion thermal velocities, but neglects trajectory bending by electric or magnetic fields. The ion density at a point is obtained by integrating the ion distribution at that point over the range of ion velocities:

$$n_i(\vec{x}) = \int f(\vec{x}, \vec{v}) d\vec{v} \quad (12)$$

The ion distribution is a function of position \vec{x} and of the ion velocities \vec{v} . The ion distribution function is redefined as the product of the unperturbed velocity distribution function and a geometrical factor as follows:

$$f(\vec{x}, \vec{v}) = \sum_{\phi=1}^{N_\phi} g(\phi_i) f_{cum}(\theta)/N_\phi \quad (13)$$

The $f_{cum}(\theta)$ array was computed in FCAL while the geometrical factor is determined by scanning along lines of constant ϕ in the x',y' plane and finding the points at which an unblocked position becomes blocked or a blocked position becomes unblocked. If the given point is behind the object, a θ can be found by interpolating for the object boundary between blocked and unblocked scans and finding the z_{ext} for this interpolated position. The angle $\theta = \arctan(\frac{r}{z_{ext}})$ where r is the distance from the interpolated boundary position to the given point.

The structure diagram for SHADO is shown in Figure 3.

The majority of coding in SHADO is devoted to determining the status of the reference point and the scan points on the hexagonal projection mesh. The status includes determinations of whether a particular point is within the hexagonal mesh or off of it; whether it is in front of the object; whether the point is blocked in the ram direction, and if blocked, whether the point is inside the object. As indicated in the structure diagram, the reference point (input) is first checked. If the point is inside the object, in front of it, or so far off to the side of the object that it would not be possible to intersect the object within the 99.9% limit, then the GI value is returned as

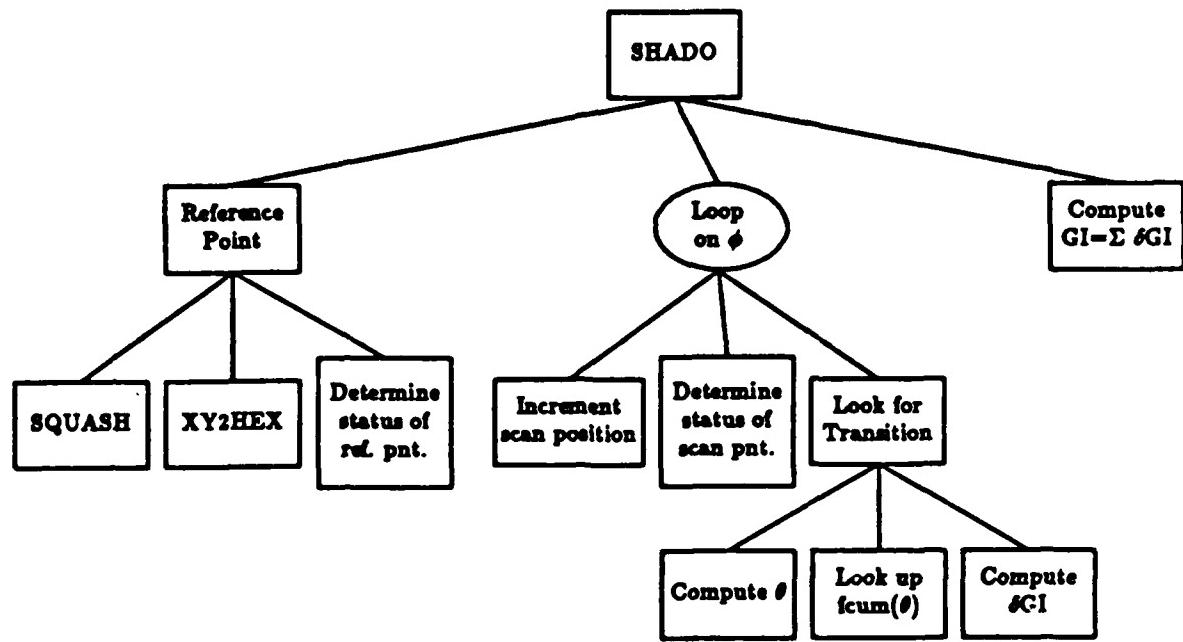


Figure 3: Structure Diagram for the SHADO subroutine

zero without further calculation.

Once the reference point status is determined, a loop is entered which initiates scans along the scan angles ϕ_i calculated in ANGLES. For each ϕ_i , a transition is sought which marks the outline of the object on the projection surface. If the reference point is not blocked with respect to the ram direction, then a transition is sought where the object first blocks a scan point. If the reference point is blocked, then a transition to unblocked status is sought.

When a transition is found, the angle θ is computed which is the angle from the reference point to the object surface at the transition point. If the reference point is blocked and behind the object, the density contribution for ϕ_i is $fcum(\theta)/N_\phi$. If the reference point is unblocked, then density contribution is $1. - fcum(\theta_1) + fcum(\theta_2)$ where θ_1 is the angular position of the first transition to blocked status, and θ_2 is the transition back to unblocked status.

References

- [1] I. Katz, D. Parks. 'A Model of the Plasma Wake Generated by a Large Object.' *IEEE Transactions on Nuclear Science*, Vol. NS-32 No. 6. December 1985, pp 4092-4096.
- [2] J. Lilley, D. Cooke, G. Jongeward, I. Katz. *Polar User's Manual*. S-CUBED Final Report AFGL-TR-85-0246, to the Air Force Geophysics Laboratory, Hanscom AFB, September 1985. ADA173758

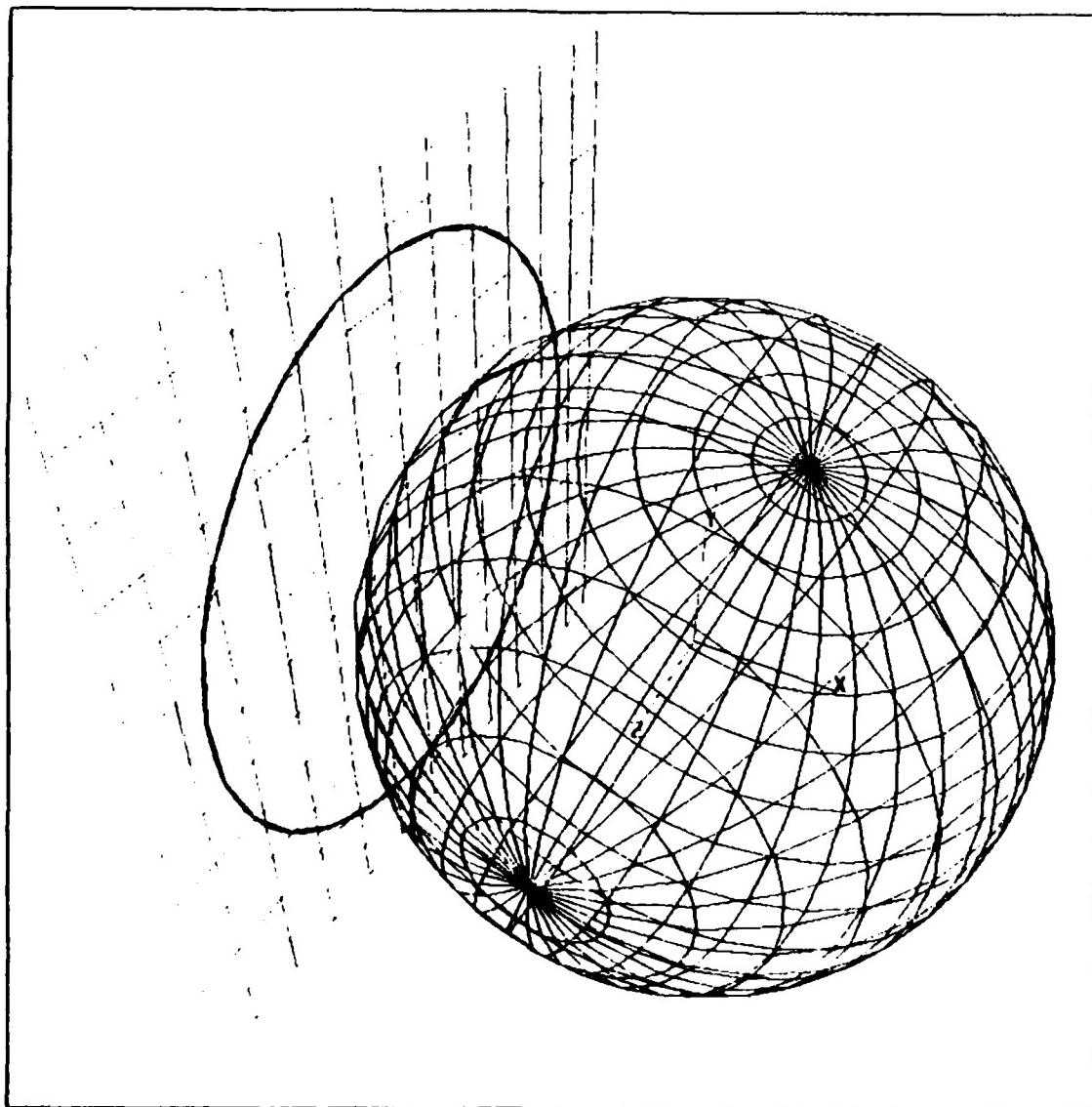


Figure 4: Projection Surface for a Spherical Object

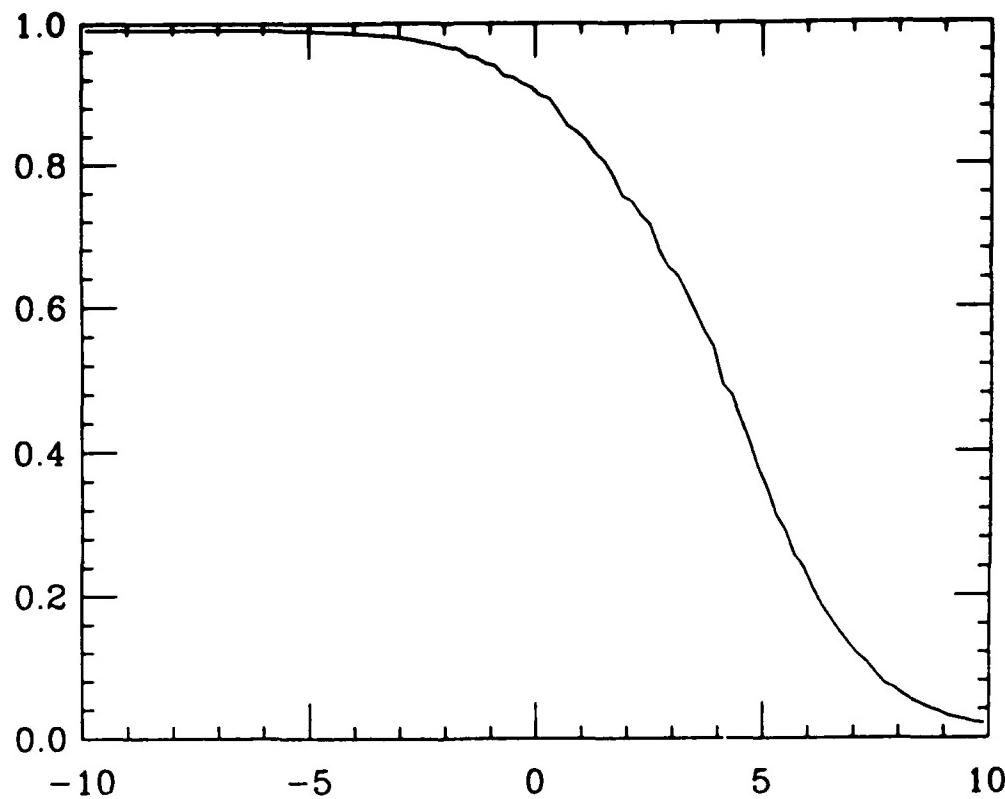


Figure 5: Quasisphere Wake results at $z/R=5$ using the old algorithm

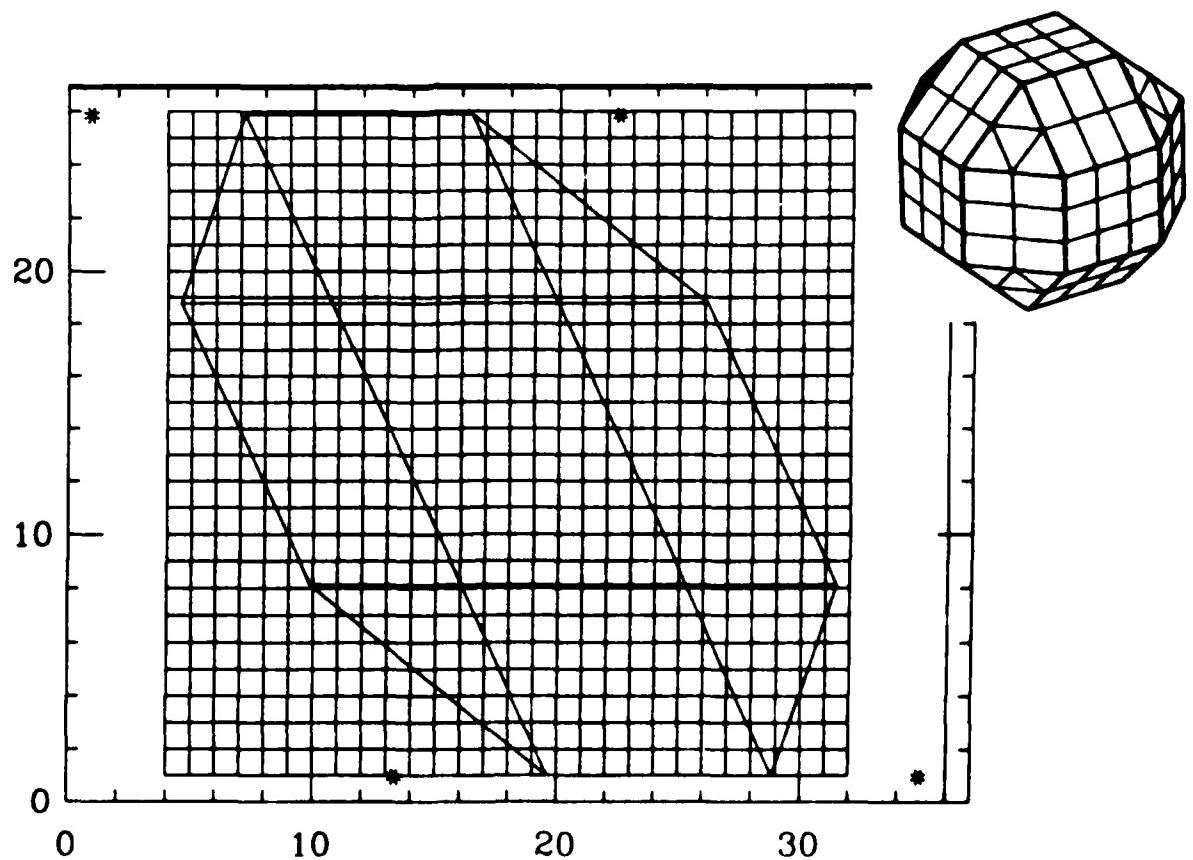


Figure 6: Projection of the quasisphere onto the skewed hexagonal mesh

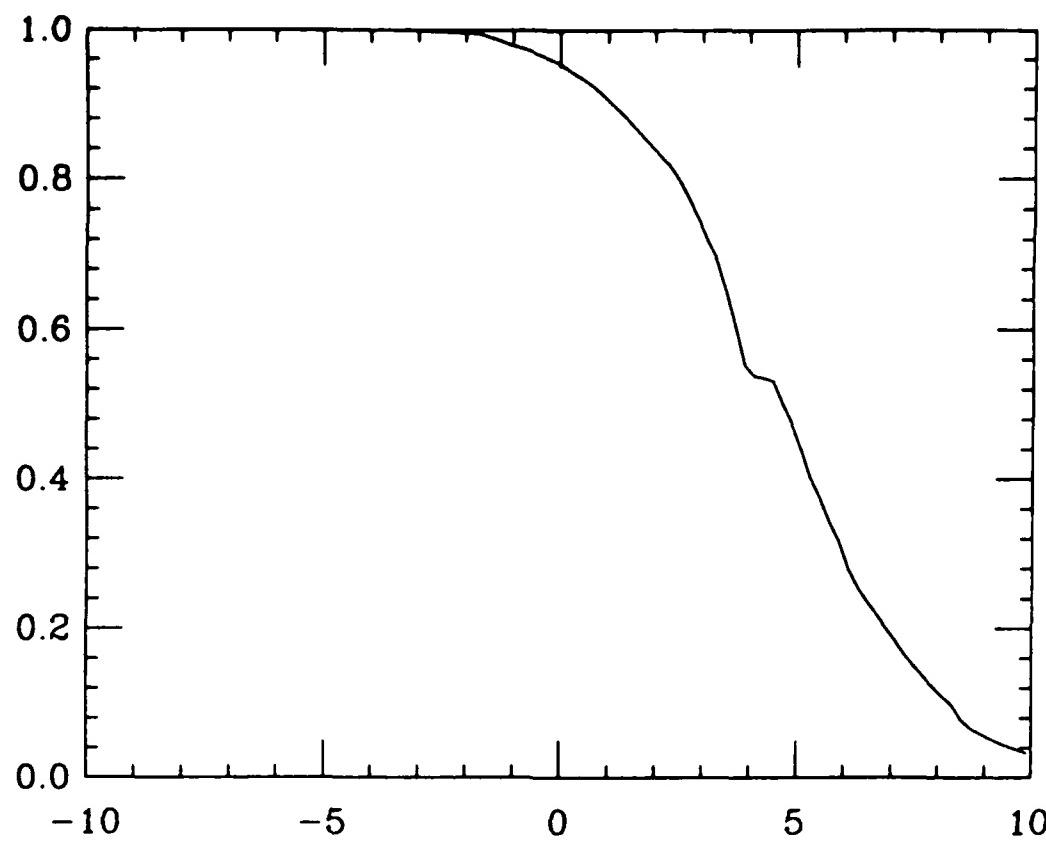


Figure 7: Quasisphere Wake results at $z/R=5$ using the SHADO algorithm with $N_\phi = 8$

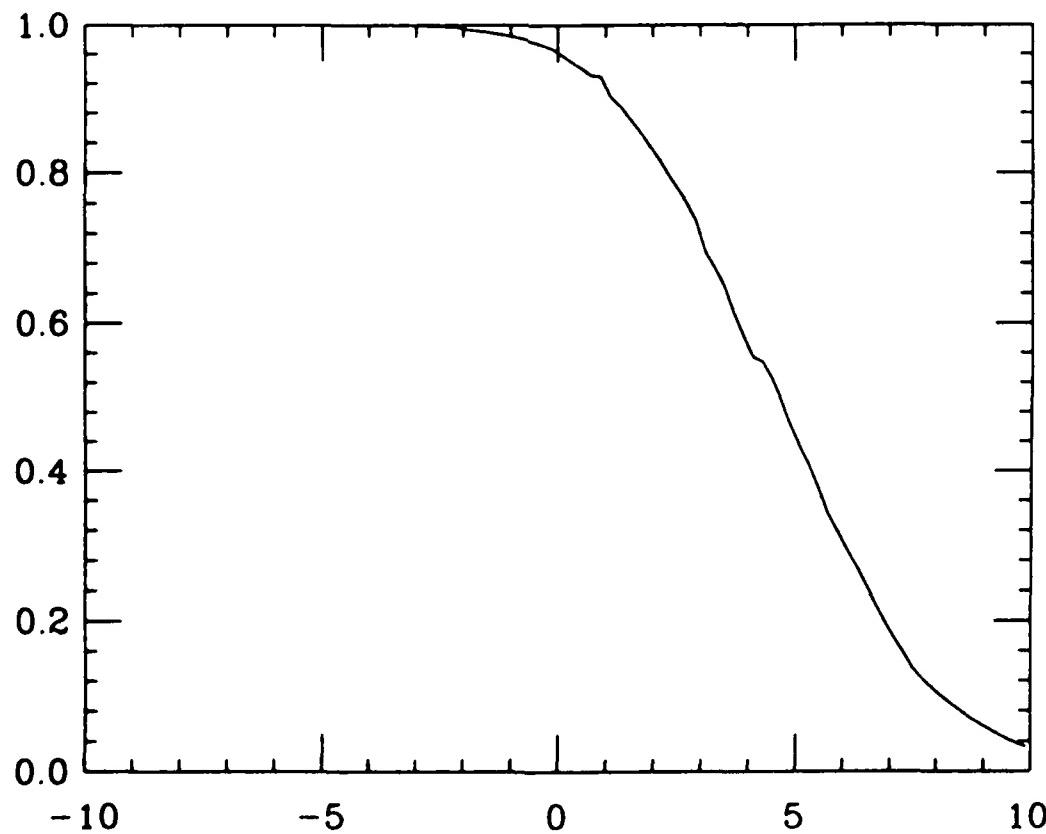


Figure 8: Quasisphere Wake results at $z/R=5$ using the SHADO algorithm with $N_\phi = 16$

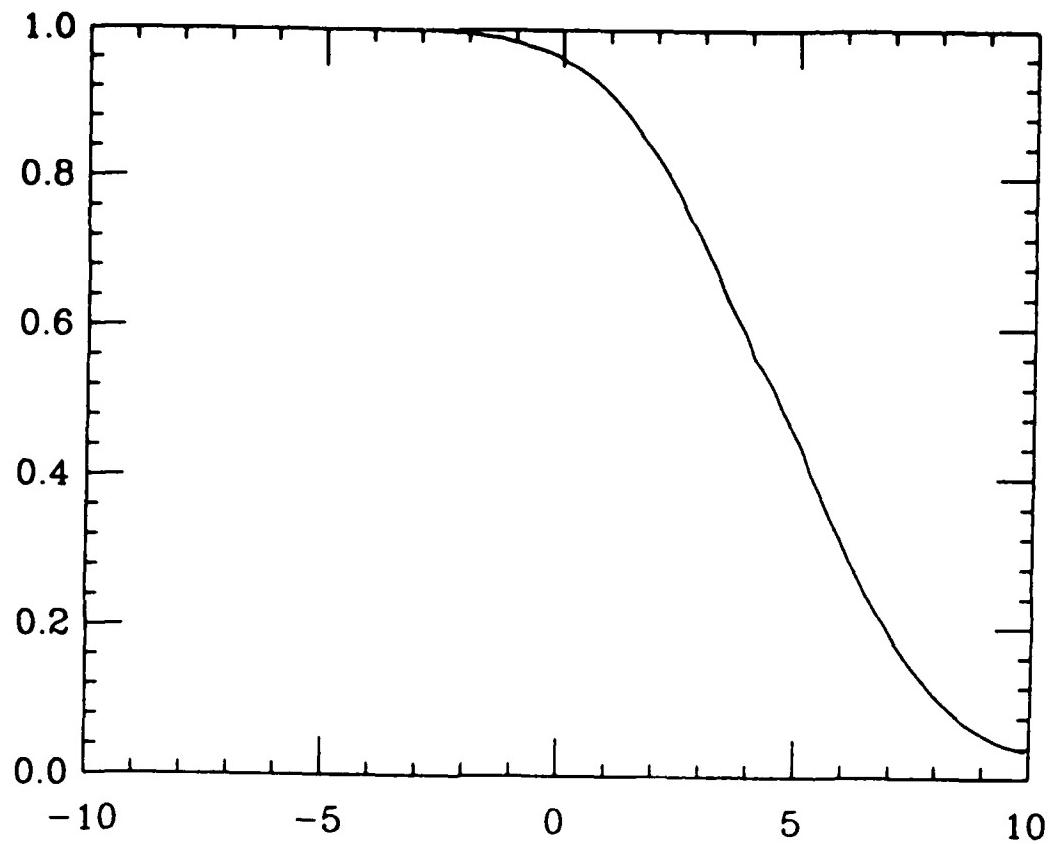


Figure 9: Quasisphere Wake results at $z/R=5$ using the SHADO algorithm with $N_\phi = 32$

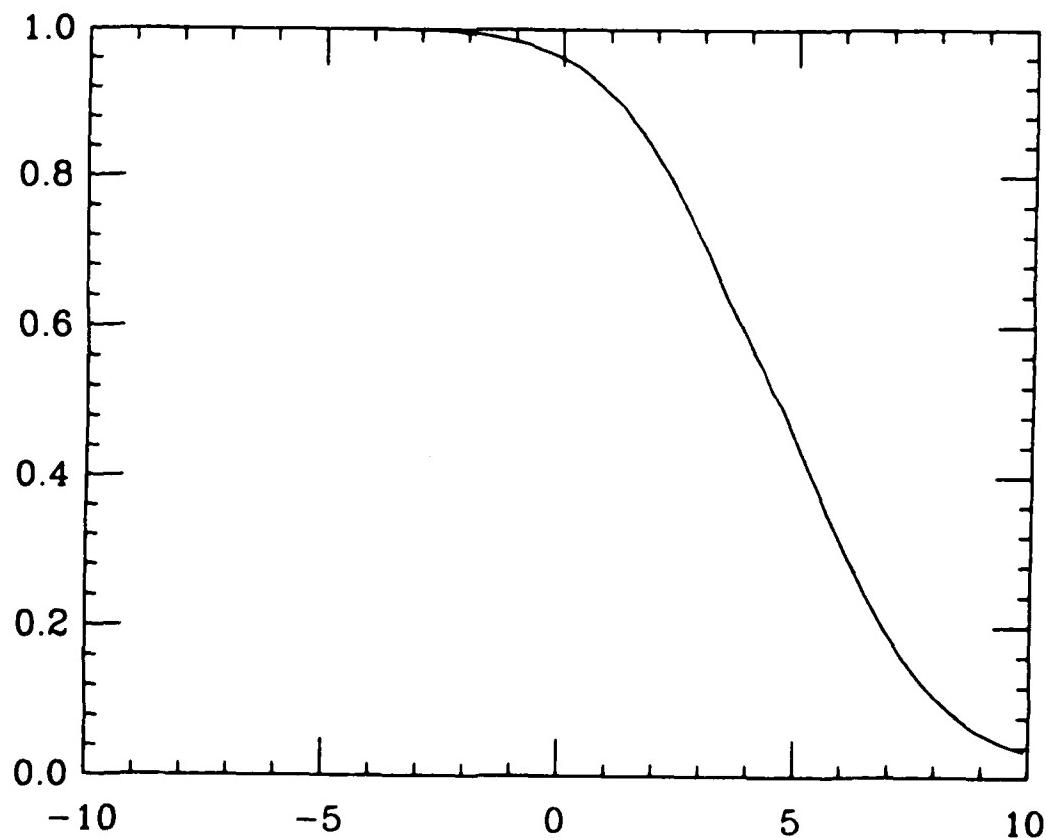


Figure 10: Quasisphere Wake results at $z/R=5$ using the SHADO algorithm with $N_\phi = 64$

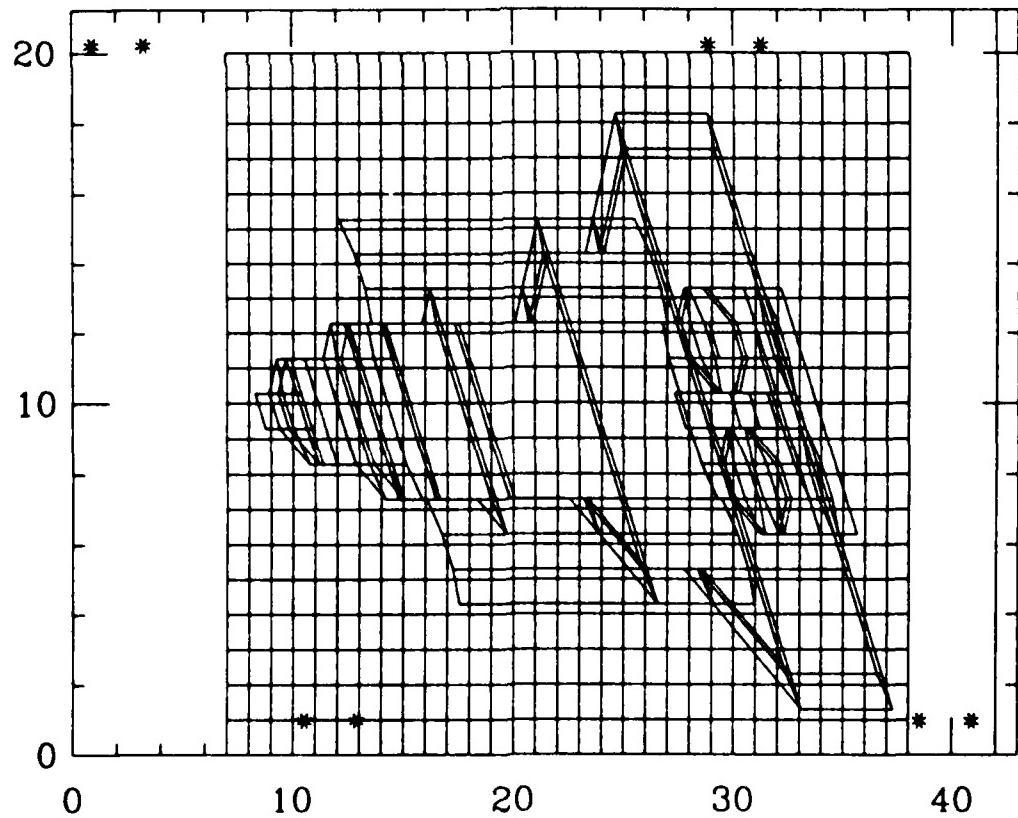
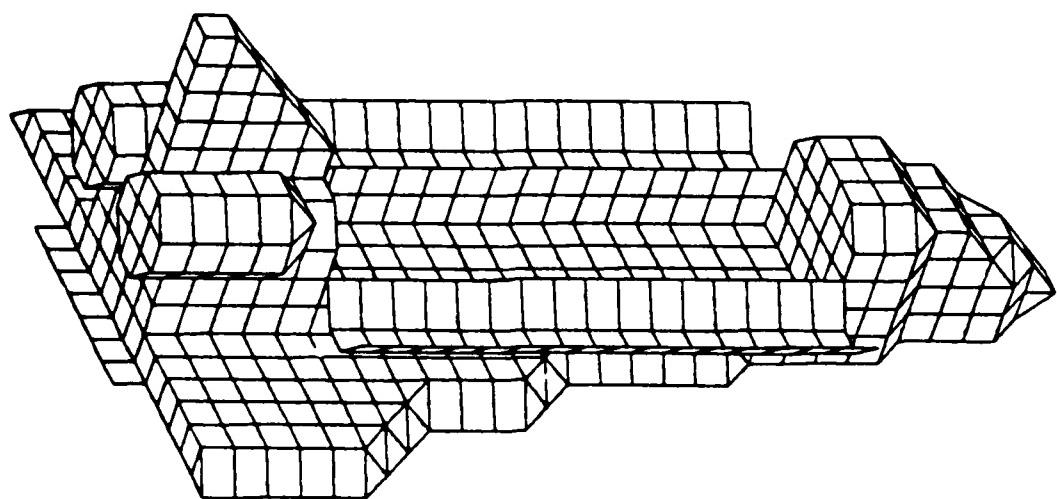


Figure 11: The shuttle object and its projection onto the skewed hexagonal mesh. The ram direction is pointing directly into the cargo bay doors.

COLOR LUT3D

9.5E-01
9.0E-01
8.5E-01
8.0E-01
7.5E-01
7.0E-01
6.5E-01
6.0E-01
5.5E-01
5.0E-01
4.5E-01
4.0E-01
3.5E-01
3.0E-01
2.5E-01
2.0E-01
1.5E-01
1.0E-01
5.0E-02
0.0E+00

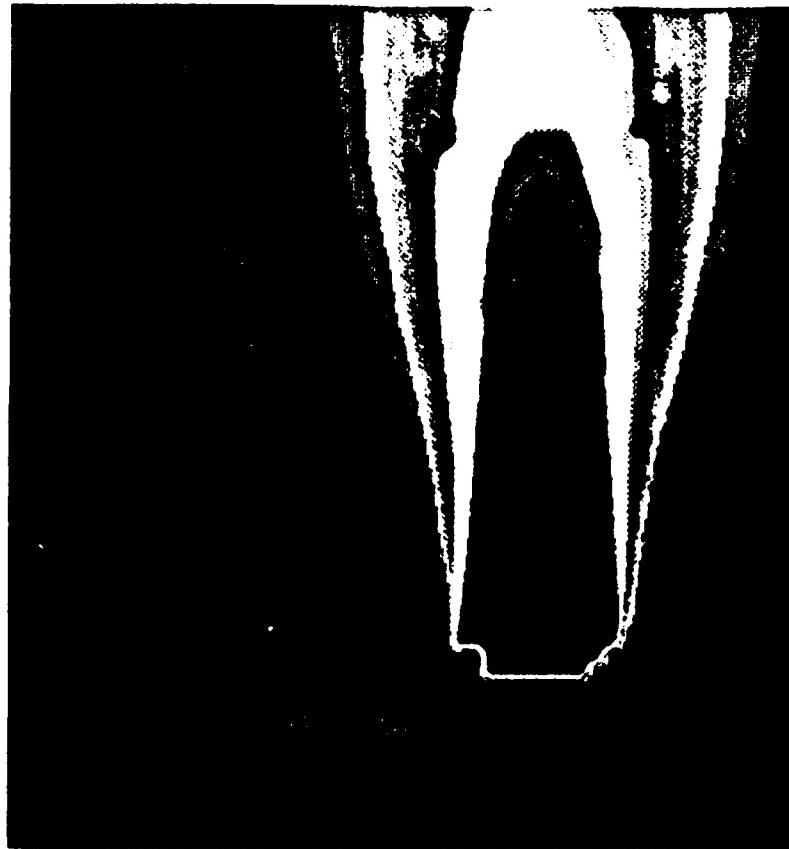


Figure 12: The particle wake behind the shuttle using the SHADO routine with $N_{hex} = 1000$ and $N_\phi = 16$

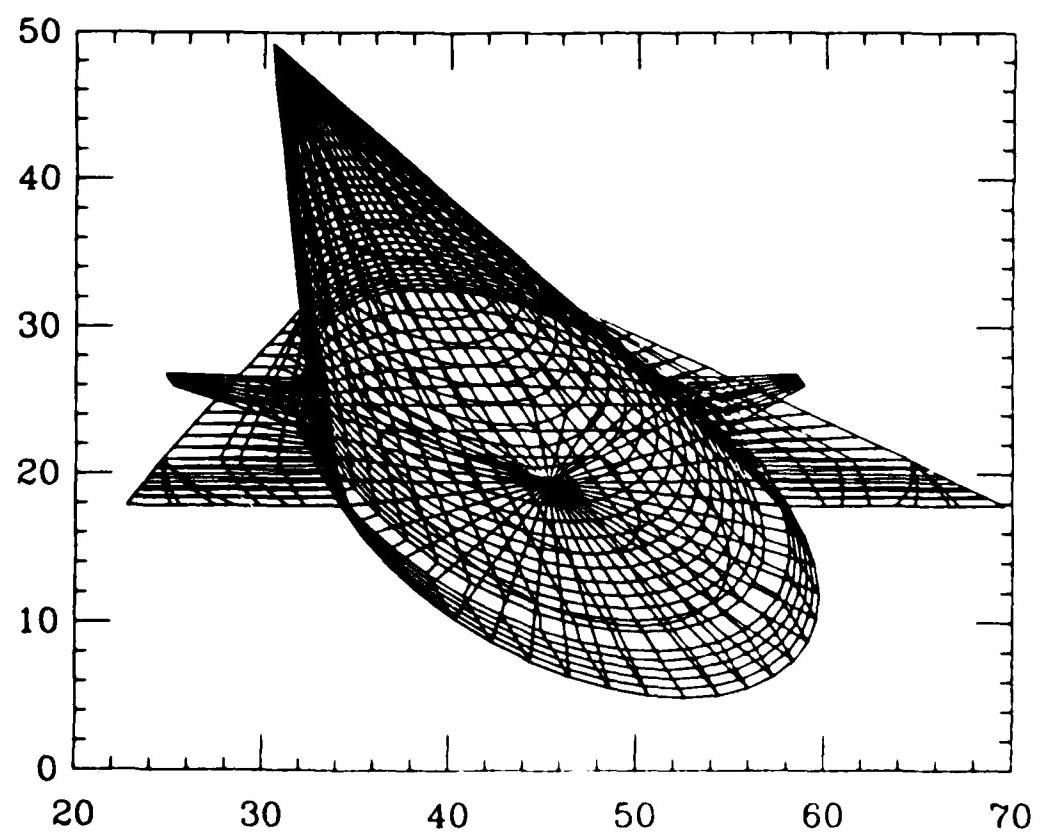
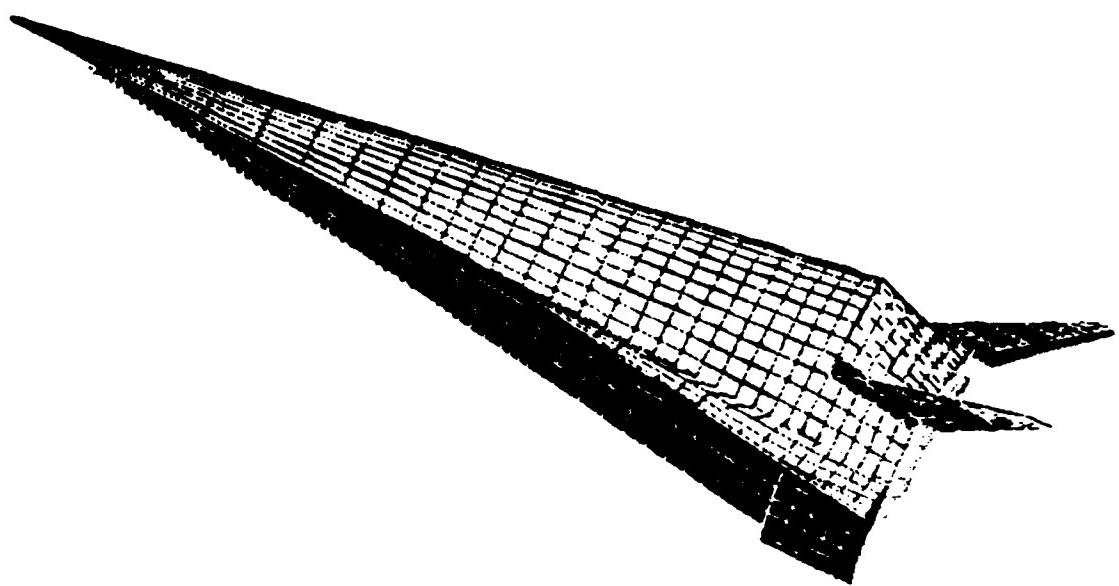


Figure 13: The aerospaceplane object and its projection onto the skewed hexagonal mesh. The plane is flying at a 20 degree angle of attack to the ram direction.

3-D Collisionless Wake Calculation

COLOR LAYER

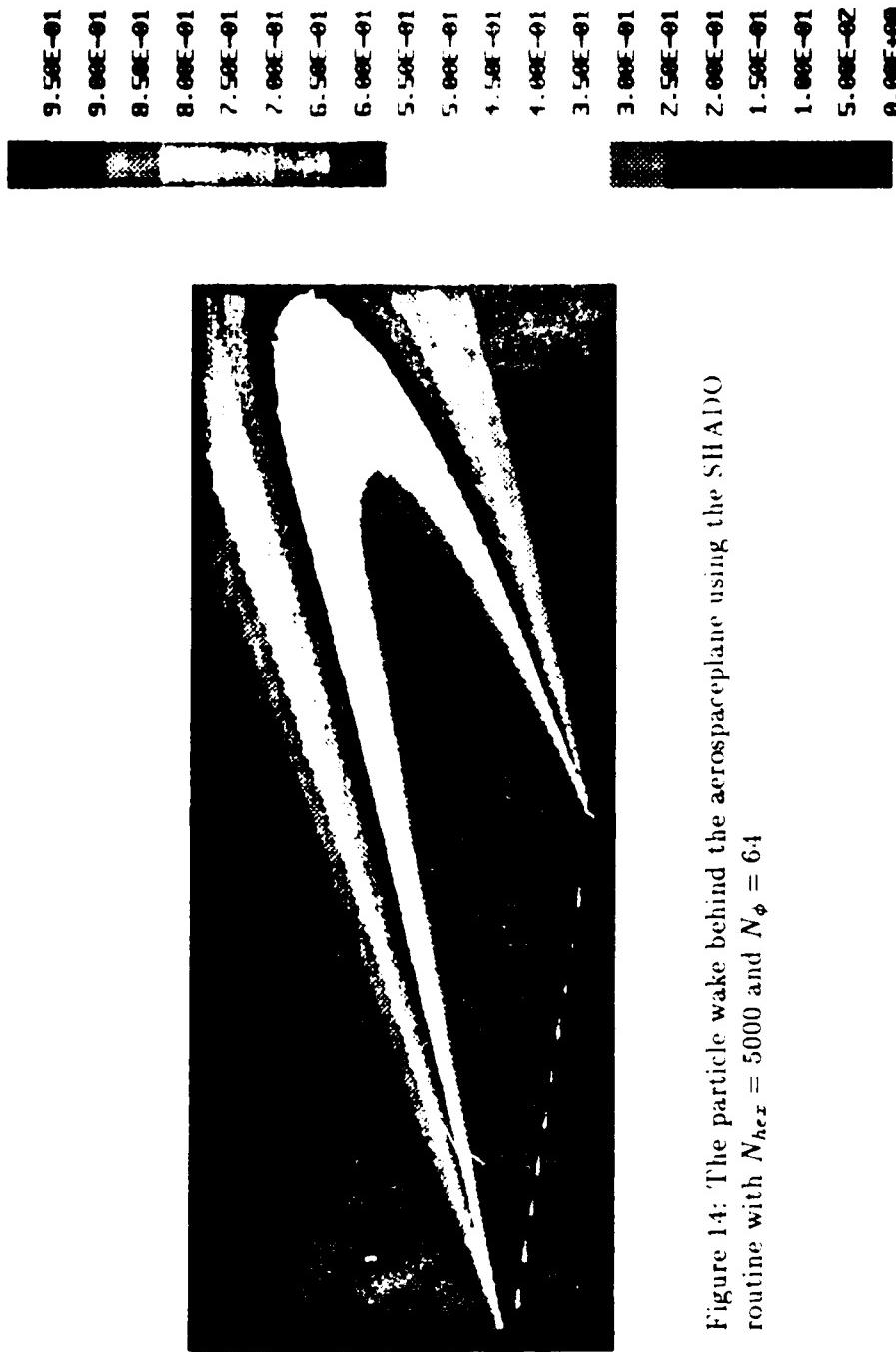


Figure 14: The particle wake behind the aerospaceplane using the SH ADO routine with $N_{hex} = 5000$ and $N_\phi = 64$

~~END~~

7 - 87

DTIC